REMARKS

Claims 1-3, 5-7, 11-14, 16-18, 27, 28, 30, and 31 are pending.

Claims 4, 8-10, 15, 19-26, and 29 have been cancelled.

In the Office Action mailed April 28, 2010, claims 1, 11-14, 16-19, 27, 29, and 31 were rejected under 35 U.S.C. § 102(a) as anticipated by Kuno ("Conversions + Interferences = Business Logic"); claims 2, 3, 5-7, 9, 28, and 30 were rejected under 35 U.S.C. § 103(a) as unpatentable over Kuno in view of Czerwinski ("An Architecture for a Secure Service Discovery Service").

REJECTIONS UNDER 35 U.S.C. § 102(a)

Independent claim 1 has been amended to further define that the run time is a time during which the node with no hard-coded conversation logic is being executed. Support for the amendment of claim 1 can be found at least in Figs. 3 and 4 and the accompanying text of the present application, including page 11, lines 6-14, and page 18, line 4 – page 20, line 2.

The amendments made to claim 1 further highlight the distinction between the claimed invention and Kuno.

It is clear that Kuno does not provide any teaching or hint of the following combination of elements of claim 1:

> b) when executing the node with no hard-coded conversation logic, dynamically discovering, by a computer, a service associated with the node with no hard-coded conversation logic, wherein the discovered service is selected from among plural services;
>
> c) selecting one of the plural conversation logic in the conversation logic repository based on the discovered service; and
>
> d) dynamically plugging in the determined selected conversation logic into the node at run time in the computer, wherein the run time is a time during which the node with no hard-coded conversation logic is being executed.

As purportedly disclosing clause d) of claim 1, the Office Action cited § 4.1 on page 10 of Kuno. 04/28/2010 Office Action at 6. Section 4.1 of Kuno refers to client automation, and explains that conversation logic is decoupled from business logic on the client side to increase the flexibility of a client by allowing the client to interact dynamically with services even if their conversation policies do not match exactly. Kuno, page 10, § 4.1, ¶ 1. To decouple conversation

logic from the business logic on the client side, Kuno discloses the use of a conversation controller to direct both the server and client sides of the conversation. *Id.*, § 4.1, ¶ 2. The process of the conversation controller is depicted in Fig. 4 of Kuno, on page 10. Step 1 in Fig. 4 of Kuno refers to looking at the message header and determining the current state of the conversation (where it appears that the message header is in a message received from a client). After verifying that the message is a valid input document type for a current state, step 4 of Fig. 4 of Kuno indicates that the conversation controller looks up the inbound document and dispatches the message to an appropriate service entry point of a service. Step 5 in Fig. 4 of Kuno indicates that the conversation controller calculates the conversation's new state. Moreover, step 6 of Fig. 4 of Kuno indicates that the output document from the service (that is being returned to the client) is formatted in a form appropriate to the client type.

As further explained in § 4 of Kuno on page 8, the conversation controller acts as a proxy to an e-service, and tracks the state of an ongoing conversation based on the types of messages exchanged. *Id.*, § 4, ¶ 1. Paragraphs 2 and 3 in § 4 of Kuno further explain that the conversation controller dispatches messages to an appropriate service entry point of an e-service, and forwards responses from the e-service to the client, with ¶ 3 of § 4 of Kuno explaining that the response sent to the client can be transformed by the conversation controller before forwarding to the client.

From the discussion of Kuno, it is clear that the conversation controller of Kuno acts as a translator or a converter between a client and a service, with the conversation controller formatting requests and responses appropriately and sending them to appropriate entry points of the client and service. The translation/conversion behavior of the conversation controller does not provide any hint of the following element of claim 1: dynamically **plugging** in the determined selected conversation logic into the node (of the workload definition with no hard-coded conversation logic) at run time in the computer, where the run time is a time during which the node with no hard-coded conversation logic is being executed. Note that the determined selected conversation logic that is dynamically plugged in at runtime is selected from plural conversation logic in the conversation logic repository based on a service discovered when executing the node with no hard-coded conversation logic.

The translation/conversion process of Kuno does not perform any dynamic plugging in of a selected conversation logic into a node at run time.

The Response to Arguments section of the Office Action further cited § 3.2, ¶ 2, on page 5 of Kuno as purportedly disclosing dynamically plugging the selected conversation logic into the node at run time. 04/28/2010 Office Action at 4. Section 3.2 of Kuno notes that a conversation developer, such as a standards body, can create a web service conversational language (WSCL) that is published in a UDDI directory. Kuno, page 5, § 3.2, ¶ 2. A software developer who wants to create an application using the published web-service can download the WSCL files describing the conversation supported, and implement the necessary methods accordingly. *Id.* It is clear that the discussion in § 3.2, ¶ 2, of Kuno relates to **software development** of an application that uses a published web-service. Accessing the WSCL files for a description of a conversation during software development has nothing to do with the subject matter of claim 1, which relates to dynamically plugging in the determined selected conversation logic into the node **at run time** in the computer, where the **run time** is a time during which the node with no hard-coded conversation logic is being executed. The software development referred to in § 3.2 of Kuno occurs before any execution of any node within a workload definition can occur.

In view of the foregoing, it is respectfully submitted that claim 1 is clearly allowable over Kuno.

Independent claim 11 has been amended in similar fashion as claim 1, and is allowable over Kuno for similar reasons as claim 1.

REJECTIONS UNDER 35 U.S.C. § 103(a)

Independent claim 3 has been amended to recite that the run-time is a time during which a node of a workload definition is being executed, where the node is with no hard-coded conversation logic. Claim 3 has further been amended to recite dynamically plugging in the selected conversation logic into the node with no hard-coded conversation logic at the run-time. Support for the amendments of claim 3 can be found in similar passages as identified above for claim 1.

It is respectfully submitted that the hypothetical combination of Kuno and Czerwinski fails to disclose or hint at all elements of claim 3. The secondary reference, Czerwinski, was cited by the Office Action as purportedly disclosing the sending of a selection query to an electronic services platform or other service broker. 04/28/2010 Office Action at 11. However,

it is clear that Czerwinski provides no hint of the dynamic plugging in of a selected conversation logic into a node with no hard-coded logic at run-time, where the run-time is a time during which a node of a workflow definition is being executed. Moreover, as explained above in connection with claim 1, Kuno also provides no hint of the foregoing subject matter.

Therefore, it is clear that claim 3 is non-obvious over Kuno and Czerwinski.

CONCLUSION

Dependent claims are allowable for at least the same reasons as corresponding independent claims. Moreover, in view of the allowability of base claims, the obviousness rejections of dependent claims have been overcome.

In view of the foregoing, allowance of all claims is respectfully requested.

The Commissioner is authorized to charge any additional fees and/or credit any overpayment to Deposit Account No. 08-2025 (10010118-1).

Respectfully submitted,

Date: July 28, 2010

/Dan C. Hu/

Dan C. Hu
Registration No. 40,025
TROP, PRUNER & HU, P.C.
1616 South Voss Road, Suite 750
Houston, TX 77057-2631
Telephone: (713) 468-8880
Facsimile: (713) 468-8883